A strategy to specify customizable interoperability maturity models for software systems

Erasmo Leite
Federal University of Bahia, Computer Science Department, Salvador, Bahia, Brazil
erasmo@gmail.com
https://orcid.org/0000-0002-4621-9388

Ana Patrícia Fontes Magalhães Mascarenhas
State University of Bahia, Department of exact and earth science, Salvador, Bahia, Brazil
apmagalhaes@uneb.br **(Corresponding author)**
https://orcid.org/0000-0002-8608-4553

Rita Suzana Pitangueira Maciel
Federal University of Bahia, Computer Science Department, Salvador, Bahia, Brazil
rita.suzana@ufba.br
https://orcid.org/0000-0003-3159-6065

Abstract

Interoperability enables transparent communication between systems and can be achieved through levels such as: syntactic, semantic, pragmatic, and organizational. The interoperability level attained depends on the participating system's maturity. Maturity models have been used in several domains to check systems' maturity according to specific aspects. However, these models have been criticized due lacking empirical validation and effective methods to help their specifications, thus harming their adoption. This work presents a strategy comprising Amortisse (mAturity Model fOR inTeroperability In Software SystEms), a maturity model to check interoperability in software systems, and the methodology for its customization and evolution. So, aiming to make Amortisse's design rationale explicit, it was developed following a methodology. In this direction, tasks, artifacts, methods, and tools related to the maturity model definition were proposed and organized as an initial methodology to support developers. When applied to real systems, Amortisse first version was able to show the system's current interoperability maturity level, the lacking requirements to achieve the desired interoperability, and evolution to a new version to meet the characteristics of a specific domain. The results of this paper show our strategy feasibility as Amortisse was able to measure the interoperability maturity of the systems, and the methodology allows the development of the first version and its maturity model evolution. We hope that sharing this maturity model version provides future contributions, improvements, and research insights.

Keywords: Interoperability, Maturity Model, Methodology, Interoperability Assessment.

Statements and Declarations: the authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

1. Introduction

Interoperability is the ability of heterogeneous systems to inter-communicate transparently and often consists of the following levels: (i) syntactic, which ensures the exchange of information among systems based on message standard; (ii) semantic, concerns on communication meaning; (iii) pragmatic, provides that systems share the same communication intention; (iv) dynamic, where systems take into account business rules; and (v) organizational (or conceptual), where systems perform all levels described previously, i.e. constraints and business rules are aligned (Braga, 2017) (Sinderen, 2010) (Wang, 2009). Some scenarios will require syntactic and semantic interoperability, while others, pragmatic and organizational, depend on the system's maturity and knowledge. While there is a need to agree on the structure and meaning of shared information, the intended effect of business-level collaboration and

alignment is equally important (Sinderen, 2010). Among several challenges for systems to provide interoperability, identifying the requirements to achieve the desired interoperability are among those.

Several works that propose solutions to support the implementation of interoperability requirements offer isolated solutions, focusing on a certain level, such as syntactic, semantic and/or pragmatic (Asuncion, 2011), (Correa, 2012), (Kerzner, 2019) and (Vivek, 2019). However, interoperability can evolve over time according to business needs or application domains. The work presented in (Santos, 2021) pointed out more than thirty interoperability types addressed in twenty-three secondary studies. Moreover, the interoperability concept itself has changed due to the widely needed collaboration among systems, things, devices, etc. The authors (Motta, 2017) argue that interoperability should be rethought to fulfil contemporary software systems goes beyond software and hardware heterogeneity, and so address the ability of things (an object, a place, an application or anything that can engage an interaction in a system) to interact for a specific purpose, considering their differences (development platforms, data formats, culture, legal issues).

Due to the constant software systems evolution it is important to provide an instrument for both, help organizations identify the current interoperability level of their systems and assist them in other levels of achievement. Maturity models can help with these issues. Maturity can be understood as a state or condition that is explicitly defined, managed, measured, and controlled in a particular discipline of an organization or domain (Correa, 2012). A Maturity Model (MM) is usually used as an instrument to support internal and/or external benchmarks of an organization's aspect as well as guidelines for future improvements. Therefore, MM has been employed to understand, plan and evolve software systems, processes and so on. A MM usually follows a stage growth approach, presenting maturity levels in a linear and unidirectional path, from the lowest to the highest (Carvalho, 2019). The higher the maturity level of an organization, the greater its services and processes quality, as well as the possibility for an organization to implement improvements in related disciplines.

Over the past few decades, there has been a growing interest in maturity models for different domains including interoperability (Carvalho, 2019), (Carvalho João Vidal and Rocha, 2017), (Lehmkuhl, 2013), such as web and social media (Duane, 2012) (Lehmkuhl, 2013), analysis (Cosic, 2012) the consulting area (Fath-Allah, 2015), industry (Ganzarain, 2016), business intelligence (Qushem, 2017), scrum (Almeida, 2017) project management (Kerzner, 2019), artificial intelligence (Messom, 2019) and logistics (Carvalho João Vidal and Rocha, 2017), (Facchini, 2020). Despite these research works, maturity models have been immersed in criticism for lack of empirical validation, misguided structural assumptions and being too simplistic to be useful leading to a poor adoption (Carvalho, 2019). Additionally, MM proposals are focused on a specific domain, and therefore, have predetermined levels, stages and requirements described in a high level of abstraction, which does not always represent the specific characteristics and organizations' needs (Rezaei, 2014), (Leal, 2019). MM's evolution to adapt to new domains and new organization requirements is not a trivial task.

This paper presents a strategy that comprises Amortise (mAturity Model fOR inTeroperability In Software SystEms), a maturity model to check interoperability in software systems, and the methodology for its customization and evolution. Amortisse has four dimensions (related to interoperability levels) and five maturity levels. It also has a set of interoperability capability indicators (e.g. requirements) to evaluate systems. Aiming to explicitly make Amortisses's design rationale, the activities performed, strategies and techniques adopted and artifacts produced were organized in the methodology (Monteiro, 2020). This methodology systematizes elements involved in MM development, such as MM domain requirements, organization of related concepts into levels, dimensions, and the path to maturity. As a result, Amortisse can be better understood, validated and customized to specific demands. So, therefore, our solution comprises the Amortisse and its maintenance and evolution methodology. While organizations may use Amortisse to measure the interoperability level of their systems and as an instrument to support software evolution in order to achieve the desired interoperability level, our methodology helps Amortisse customization and evolution.

We evaluated Amortisse through two case studies. Besides, we also evaluate the user perception concerning aspects such as usefulness and consistency.

We hope that sharing our proposal with the community for future improvements contributes to MM developers and users toward maturity model adoption as an interoperability achievement.

Besides the introduction, this paper is organized into the following sections: Section 2 introduces the underlying concepts of this work and State of the art; Section 3 describes the maturity model development methodology; Section 4 describes the Amortisse; Section 5 describes the Amortisse assessment; and finally Section 6 presents concluding remarks and opportunities for future work.

2. State of Art

This section describes some concepts related to maturity models (Section 2.1) and interoperability (Section 2.2). Additionally, analyzes current solutions (Section 2.2.1) concerning maturity models for the interoperability domain.

2.1. Maturity Models

In general, maturity can be defined as "the state of being complete, perfect or ready" (Lahrmann, 2011). Maturity thus implies evolutionary progress in the demonstration of a specific ability or the accomplishment of a target from an initial to a desired or typically occurring end-stage. In the Information System discipline, maturity is instead regarded as a measure to evaluate the capabilities of an organization (de Bruin, 2005). Following Becker et al. (Becker, 2009), MMs endorse this evaluation by outlining anticipated, typical, logical, and desired evolution paths. Furthermore, Mettler and Rohner (Mettler, 2009) argue that, as formality is incorporated into the organizational development activities, decision-makers are given a pragmatic instrument to determine whether potential benefits have been realized or not.

A Maturity model usually encompasses five important components:(i) *maturity stage* also known as levels or maturity score, and it describes the main goal of the maturity in a stage; (ii) *dimension*, which represents critical factors for success in the analyzed domain, and may contain subcategories (Carvalho, 2019); (iii) *subcategory*, represents second-level variables on which a dimension depends on. A subcategory groups requirements related to a specific subject of the evaluated entity; (iv) *requirement* or assessment question represents a necessity of the domain under evaluation that should be reached. It is usually directly linked to the sub-categories with a maturity score or stage; and (v) *maturity assessment follows*, a path (to maturity), which usually is linear, from the lowest to the highest maturity stage. Moreover, maturity models usually have two parts: (a) a generic design structure comprising of different stages, each with different dimensions and sub-categories; and (b) hierarchical relationships between the typical components of the maturity model. MMs can be seen as an established means to systematically document and guide organizations' development using capability stages.

MM can be specified using two different approaches: top-down or bottom-up (de Bruin, 2005). In a top-down approach, a fixed number of maturity stages is initially specified and further corroborated with some characteristics (i.e., assessment items, e.g., requirements) that support first assumptions about the maturity distribution into dimensions and subcategories (Lahrmann, 2011). In a bottom-up approach, distinct characteristics are first determined and then clustered into maturity stages.

MMs have proliferated across a multitude of domains since the concept of measuring maturity was introduced by the Capability Maturity Model (CMM) from the Software Engineering Institute (SEI) – Carnegie Mellon (de Bruin, 2005). The SEI has created six maturity models in total and has recently incorporated three legacy CMMs into one maturity model, now named the Capability Maturity Model Integration – CMMI (Chrissis, 2011). The CMMI has motivated consultants to develop some models for several proposals, thereby increasing their adoption (Chrissis, 2011).

Whilst maturity models are high in number and broad in application, there is little documentation on how to develop a maturity model that is theoretically sound. Limited knowledge is available on how to delineate the MM path evolution systematically and how to instantiate the corresponding MM (Brocke, 2019). As for now, few distinct development processes have been discussed in current literature to assist MM development, e.g. in (Becker, 2009) and (de Bruin, 2005). One standard development framework forms a sound basis to guide the development of a model and then enable the evolution of the model. Furthermore, whilst decisions within the phases of this framework may vary, the phases themselves can be reflected in a consistent methodology that can be applied across multiple disciplines.

2.2. Interoperability

Interoperability has become a critical non-functional requirement as distributed computing, web, and service-oriented architecture have overcome organizational barriers, connecting previously isolated business data (Neiva, 2016). Interoperability is commonly related to application collaboration, regardless of the technologies used, e.g. methods, programming languages, and environments.

The Levels of Conceptual Interoperability Model (LCIM) is the framework for assessing the degree of conceptual representation between interoperable systems, which classifies interoperability into seven levels: no interoperability, technical, syntactic, semantic, pragmatic, dynamic, and conceptual (Fewell, 2003). Since LCIM, several authors have categorized interoperability into levels. For example, the work proposed in (Santos, 2021) adopts LCIM as a reference and includes a new interoperability level, the Organizational. Figure 1 illustrates some common interoperability levels 0 - stand-alone systems that have no interoperability (Correa, 2012); 1 - syntactic interoperability, which introduces a common structure to exchange information, i.e. a common data format is applied (Correa, 2012); 2 - semantic interoperability, when a common information exchange reference model is used (Correa, 2012); 3 - pragmatic interoperability, when the interoperate systems are aware of the methods and procedures that each system is employing (Correa, 2012); 4 - organizational interoperability, encompasses business rules and organizational policies and can be focused on process modelling and realigning information architectures of organizations with objectives and strategies that promote collaborative work (Correa, 2012); 5 - dynamic interoperability, considers that a system operates on data over time and the state of that system will change. This includes the assumptions and constraints that affect its data interchange (Tolk, 2007); finally, 6 - conceptual interoperability, where systems are completely aware of each other information, processes, contexts, and modelling assumptions (Tolk, 2007).

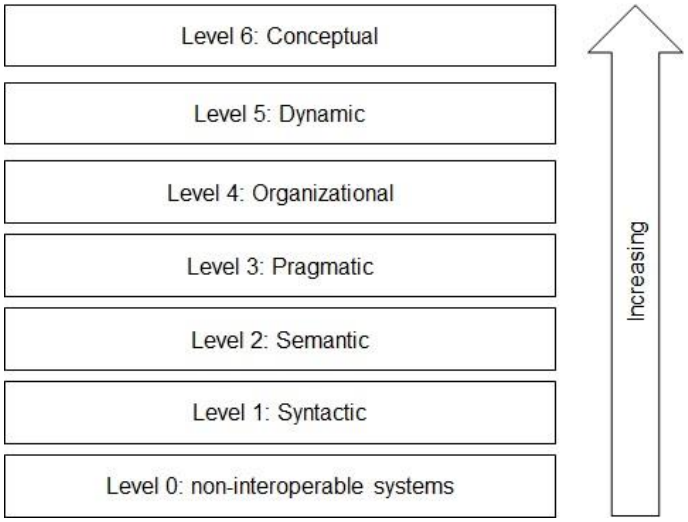

Fig. 1 Common Interoperability Levels (Santos, 2021)

For instance, some systems may require syntactic and semantic interoperability, while others may require the organizational level. Several problems occur when exchanging information among heterogeneous systems, such as programming language differences or ambiguity in the development environment. Full interoperability is the capacity of a system to achieve all desired interoperability levels from the most basic to the most specific (Boscarioli, 2017).

Strategies to solve both syntactic and semantic interoperability are quite stable in the literature. A standard usually performs syntactic interoperability, and semantic interoperability is addressed by ontology or thesaurus (Wang, 2009). Several researchers deal with pragmatic interoperability (de Moor, 2005), (Lee, 2007), (Liu, 2014), (Neiva, 2016), (Evripidou, 2007), (Ribeiro, 2019). Although both syntactic and semantic levels have consensual definitions, neither pragmatic, dynamic, nor organizational interoperability

have a commonly shared one. The increasing amount of work in the area potentiates both confusion and ambiguity in the use of these terms, making it difficult to have a comprehensive view of related or similar interoperability proposals and, so, hindering information systems from achieving the desired interoperability. The authors of (Santos, 2021) have pointed out more than twenty-seven different names of interoperability. The lack of common understanding of interoperability levels can result in (i) incompatible solutions, (ii) ambiguous interpretations of messages, or (iii) difficulty in achieving the desired interoperability.

As Interoperability is one of the aspects that can be measured in an organization's software systems, over the past few decades, there has been a dramatic interest in maturity models for different specific domains (Carvalho, 2019). However, Interoperability can evolve according to business needs, and maturity models can be an instrument for both, helping organizations identify the current interoperability level of their systems and assisting them in other levels of achievement.

2.2.1. Interoperability Solutions

There are some proposals concerning the inter-communication among systems. Recently, pragmatic interoperability, as an LCIM intermediate level, has attracted attention of a lot of works, such as (Evripidou, 2007), (Lee, 2007), (Liu, 2014), (Neiva, 2016) and (Ribeiro, 2019). In addition, many existing maturity models have been developed for different purposes, such as (Clark, 1999), (Fewell, 2003), (Chen, 2013), and (Rezaei, 2014).These proposals usually focus on a specific interoperability level.

The LISI (Levels of Information System Interoperability) maturity model defines interoperability between information systems (Clark, 1999). It focuses on technical interoperability, i.e. the ability to share information using Ethernet or fiber net, and the complexity of interoperations between information systems. LISI comprises five levels: Isolated, Connected, Functional, Domain and Enterprise. LISI enables building an interoperability matrix to represent the potential for each system to interoperate with others and displays the level at which the interactions will potentially occur.

The OIM (Organizational Interoperability Maturity Model) deals with the ability of organizations to interoperate (Fewell, 2003). It extends the LISI model to assess organizational issues at the business company interoperability level. OIM comprises five levels: Independent, Ad hoc, Collaborative, Integrated and Unified. OIM covers business areas focusing on organizational issues in an enterprise context, and it does not address technical, semantic or syntactical aspects. The EIMM (Enterprise Interoperability Maturity Model) (Chen, 2013) aims at evaluating enterprise modelling/model maturity and covers data, service and process interoperability issues.

The authors of (Rezaei, 2014) present a maturity model for the interoperability of ultra-large-scale systems, a new generation of distributed software systems composed of various changing, inconsistent or even conflicting components distributed in a broad domain. Some important characteristics of these systems include their considerable size, global geographical distribution, and operational and managerial independence of their member systems. By using the interoperability level of the component system of one ultra-large-scale systems, its maturity level can be determined. By proposing a framework, the authors tried to increase the interoperability of the component systems in ultra-large-scale systems based on the interoperability maturity levels. Consequently, their interoperability is improved.

SPICE (Software Process Improvement and Capability Determination) is a general model for processes maturity (Guédria, 2015). Although it is not designed specifically for interoperability, it is important because a higher maturity of enterprise processes may contribute to developing interoperability between enterprises. SPICE can also be used to evaluate the maturity of this process. Additionally, this paper presents a comparative showing among some interoperability maturity models (LISI, OIM, LCIM, EIMM and in (Rezaei, 2014)). It concludes that they are partial models only dealing with some aspects of the enterprise interoperability domain.

The Interoperability Maturity Model (IMM) (Knight, 2020), proposed by the U.S. Department of Energy (DOE), is a MM for the information and communications technology integration of intelligent devices and systems in various electric power system technology domains, such as interactions with the bulk generation, transmission and distribution infrastructure, and distributed energy resources in customer systems. IMM comprises 32 criteria, which are grouped into six categories for quantifying the state of interoperability in

a technology integration domain. The six categories are related to interoperability levels: technical, informational, organizational, operation and performance, safety and security, configuration and evolution. It comprises physical device aspects and the technical interoperability categories comprise basic, network and syntactic connectivity aspects. Each criterion has five levels of descriptions (initial, managed, defined, planned and optimized) that can be used to assess interoperability maturity related to the categories. The first step in measuring interoperability is identifying a target integration situation and the interfaces that support it. Once a target for applying to the IMM has been selected, it is necessary to decide whether to apply for the whole IMM or part of it. The choice of parts (categories) to use may be driven by known interoperability deficiencies or specific drivers that cause the stakeholder(s) to prioritize one category over another. The report presents an example of IMM use, but no formal evaluation is provided.

Except for IMM, each of these maturity models aims to assess interoperability through different perspectives, approaches and metrics, but using a linear path to maturity without considering equifinality, i.e. an entity or system can achieve the same result from different initial conditions and through different paths. However, while IMM levels and stages are fixed, Amortisse structure (level, stage, and requirements) can be customized, for instance: a new level representing a distinct interoperability type can be added. Additionally, Amortisse is domain independent and IMM target is the electric power system domain.

Amortisse provides an inductive form of structuring dimensions and levels, making it possible to guide oneself through different paths among its dimensions and levels. Furthermore, interoperability differs among domains and presents an important evolutionary process, leading to the need for maturity models to evolve together. For this purpose, Amortisse was specified using a pre-defined methodology which can be applied to the Amortisse needs for evolution and adaptation to new situations faced by the systems and software of a given organization. Furthermore, the capabilities and requirements used in those related works could be added to Amortisse to fulfil specific domain aspects. Sections 3 and 4 present the methodology and the Amortisse MM, respectively.

3. Maturity Model Development Methodology

This section introduces the MM methodology adopted in Amortisse specification (Monteiro, 2020).

This methodology emerged during the specification of Amortisse, a maturity model for the interoperability domain. We found a lack of theoretical foundation in the (qualitative) approaches usually adopted in MM definitions (Santos, 2021), i.e. information about MM specification was scattered in the literature and usually described at a high abstraction level, without an appropriate level of detail to be used. So, we analyzed different maturity models aiming to identify the involved elements (e.g. dimensions and stages), selected strategies to define these elements and defined artifacts to document them. In this process, we also documented all the performed activities. The results of this analysis were systematized in this methodology. The methodology was also evaluated through Amortisse validation in real scenarios (Section 5) and can be constantly improved, as the construction or customization of MMs may require new definitions, e.g. new requirements might emerge.

Figure 2 shows an overview of the methodology life cycle, which comprises six phases: Scope, Design, Model Content, Model Construction, Implementation, and Maintenance. The methodology adopts the bottom-up strategy, so the maturity model components are first identified, and the stages are defined.

Each phase is composed of tasks to be performed toward the model specification. For example, the Scope phase aims to define the model scope and comprises two tasks, set domain and set components. Moreover, phases are sequentially executed, i.e. the artefacts produced in the Scope phase are required to perform the Design phase and so on.

MM specification starts in the *Scope* phase. Its main goal is to properly specify the model scope for a specific domain, necessary to determine outer boundaries for the model usage and possible extensions. This phase comprises two tasks: *Set Domain* and *Set Components*. The *Set Domain* task consists in understanding the domain by applying one or more methods for knowledge acquisition, such as literature reviews, interviews and so on. Although a systematic literature review is not mandatory, it can give a deep understanding of historical and contemporary domain issues. As part of this task, stakeholders are also identified. They will participate in the rest of the methodology phases specifying and validating the MM. In the next task, *Set Components*, the domain components (or categories) and sub-components (or

subcategories) are obtained. Components represent the domain concepts used to classify requirements (identified in the third phase). The stakeholders identify them according to the domain necessities. For example, in the interoperability domains important concepts, such as *network*, *security* and *database* can be mapped as components of the model. Each of these will represent a set of associated requirements.

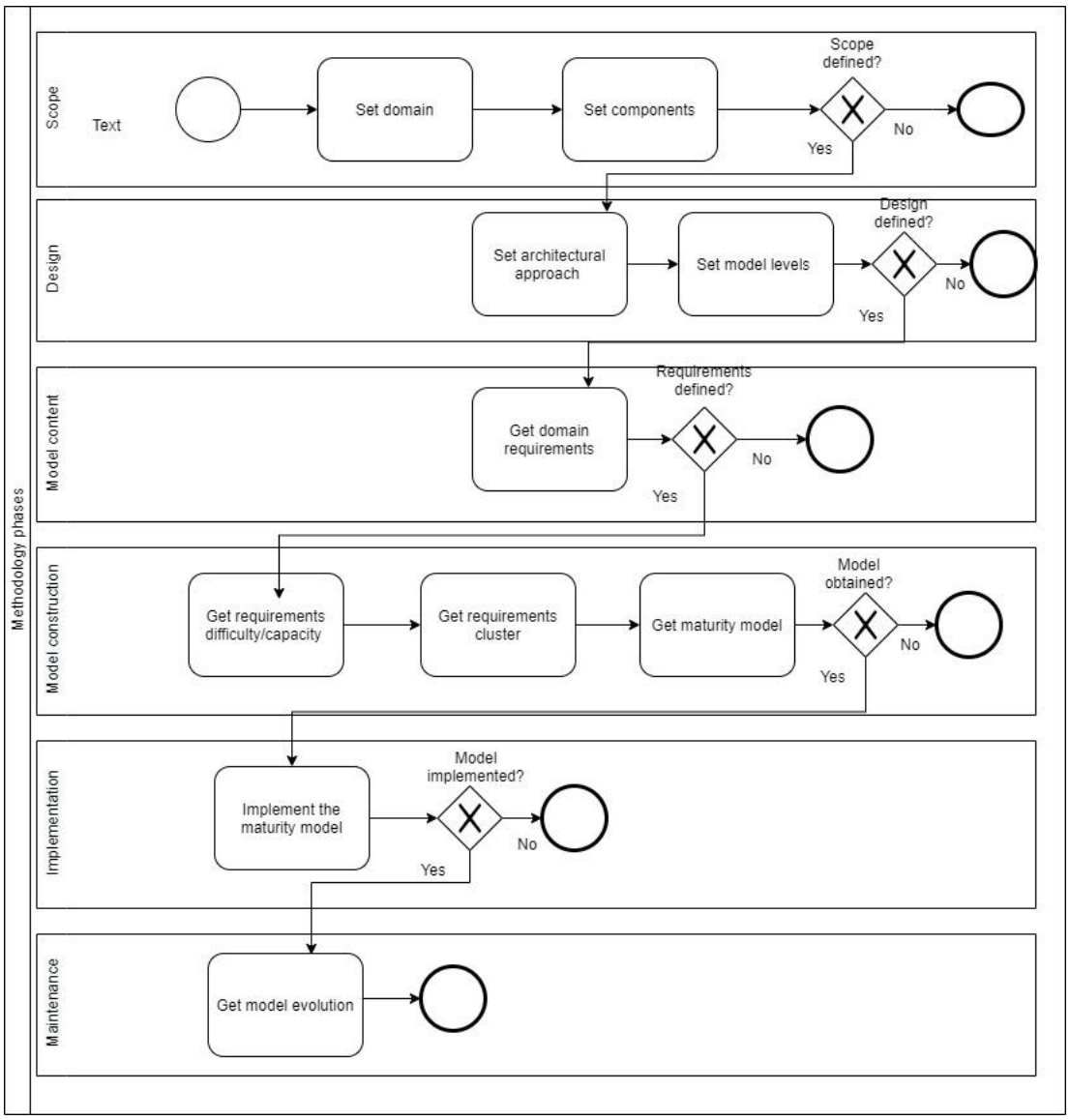

Fig. 2 Proposed life cycle: Business Process Model and Notation (Messom, 2019)

The second phase is named *Design* and aims to specify MM architecture, i.e. models dimensions and stages (or levels). The first task in this phase, the *Set architectural approach*, is to define what approach should be taken in the maturity stages to delineate the evolution path. Our methodology recommends adopting a top-down or bottom-up approach depending on the domain features (de Bruin, 2005). With a top-down approach, definitions are written first, and then measures are developed to fit the definitions. With a bottom-up approach, requirements and measures are determined first and then definitions are written to reflect these. A top-down approach works well if the domain is relatively new and there is little evidence of what is thought to represent maturity. Whichever approach is chosen, uncertainties related to the evolution path must be considered (Mettler, 2009), especially when dealing with new domains or domains in constant change. For example, when specifying an MM for an emerging domain, maturity stages may be uncertain, as the concepts are still unstable.

To define the maturity stages, qualitative methods are often adopted, for example, standardized questionnaires to assess the requirements level of difficulty. Considering this, some authors (Dekleva, 1997) recommend the use of Item Response Theory (IRT) (De Ayala, 2018) to support the definition of MM

maturity stages. This theory aims to improve the reliability and validity of standardized questionnaires and tests that conclude item difficulties or participants' skills. An example is the Rasch algorithm-based approach (Lahrmann, 2011) and the hierarchical cluster analysis (Field, 2013). So, using the selected approach, the MM stages are defined (Set model levels task) e.g., based on methods adopted in building maturity model constructs (conceptual, qualitative and so on). In existing maturity models, a common design principle is to represent maturity as several cumulative stages expressing a logical progression through the stages, i.e., the highest stages are based on the requirements of the lower stages. This practice was popularized by the CMM and appears to have broad practical acceptance. Besides, stages should be named with short labels that give a clear indication of the internship's intent. The number of stages may vary from model to model, but what is important is that the stages are distinct and well-defined.

Once the scope and design of the model are agreed upon, it is time to define the Model Content in the third phase (*Model Content*). In this phase, it is necessary to specify the requirements of each domain component (or class or category) identified in the Scope phase. These requirements are important to determine which are the maturity assessment of them and how this can be measured (task *Get domain requirements* in Figure 2). For example, syntactic interoperability can be measured through network requirements and data exchange in the interoperability domain. Therefore, the outputs of this phase are the requirements obtained for each component (or class or category) of the Scope phase and their eligibility, i.e. whether a particular requirement should be considered in the scenario. A short questionnaire can be designed to aid stakeholders in this phase.

The next phase, *Model Construction*, aims to get a first draft maturity model. The model has to be populated by setting the level of each requirement in its respective dimension. We recommend the use of a pilot study to set up what is the effort to implement a requirement in a given scenario. During the pilot study, it is possible to find the difficulty or capacity (*Get requirements difficulty or capacity* task) of each requirement and build a hierarchy of clusters (*Get requirements cluster* task) to generate the maturity model (*Get maturity model* task).

In the *Implementation* phase, the model must be made available for use to verify its extent to generalizability (*Implement the maturity model* task in Figure 2). Once the model has already been developed and tested, for example, in the pilot study of the previous phase, the initial MM (draft version) should be used in a real scenario to evaluate the maturity of something (e.g. a development process, a system, among others) from another stakeholder. The system maturity level indication is an expected result. Different methods can be used here, like brainstorming with specialists in a survey.

The *Maintenance* aims to aid model maintenance and further development when, for example, some model elements get obsolete, new requirements emerge, and assumptions on the different maturity stages are affirmed or refuted. Therefore, it is essential to reflect on how to handle alterations in model design and deployment even in an early stage. A repository can support this task. As an input for this phase, we can consider resources, such as domain requirements, necessary to keep up the model's growth and use.

Table 1 summarizes the artifacts of each methodology phase.

Tab. 1 MM Methodology artifacts

Methodology Phase	Input Artifacts	Output Artifacts
Scope	Results of a literature review(or other knowledge acquisition method)	Model scope Stakeholders name Categories and subcategories
Design	Maturity Model structure (CMM-like, Likert-like, questionnaires, Maturity matrix or grid)	Maturity model architecture (dimensions and levels)
Model Content	Components, dimensions, questionnaires results	Domain requirements classified according to the components Requirement eligibility
Model Construction	Dimension, requirements and levels	Draft maturity model
Implementation	Draft maturity model	System maturity level indication Validated maturity model
Maintenance	Resources, e.g. unit of analysis	Updated maturity model

The first column (Table 1) shows the phases and the others the input and output artefacts consumed and produced in them. For example, during the Scope phase, the results of a literature review (or other methods of knowledge acquisition) are used as a reference to define the model scope, identify the stakeholders and specify model categories and subcategories.

4. Amortisse (mAturity Model fOR inTeroperability In Software SystEms)

This section presents Amortisse, which aims to assess the interoperability level of given software systems, i.e. its current interoperability maturity. Organizations can use Amortisse to identify what requirements their systems should fulfil to achieve the desired interoperability dimension, i.e., Amortisse can assess the effectiveness of systems interoperability, figuring out what capabilities they should acquire to improve their performance. When applying Amortisse, approaches like focus groups or brainstorming can be used by team leaders to indicate which requirements are met or are missing by a system and also the effort to obtain them.

Amortisse is structured in dimensions and levels (Figure 3). While a dimension identifies the interoperability desired to measure in a system, a level measures the system maturity within each dimension. Each dimension is composed of measurable requirements (capability indicators) that make it possible to collect evidence to assess the interoperability level that a system achieves in that dimension.

In this first release, Amortisse dimensions are related to the currently most common interoperability levels found in the literature: syntactic, semantic, pragmatic and organizational, and Amortisse considers the web service domain. The web service domain was chosen because the interoperability of syntactic and semantic definitions and solutions is relatively stable.

Each dimension comprises five maturity levels (stages) that define the interoperable system achievement. Stage one is the lowest degree of interoperability, and five is the highest, i.e. most significant ability to interoperate. Figure 4 shows Amortisse levels: Stage 0 represents a lack of ability to interoperate; Stage 1 represents some knowledge about the requirements needed to interoperate; Stage 2 represents shared knowledge of what needs to be done to interoperate; Stage 3 represents an alignment of knowledge and systems to interoperate; Stage 4 indicates a representation of processes and systems in workflow; and Stage 5 represents knowledge of the system's weaknesses and processes to evolve.

Each requirement will be assessed based on a rating scale, where N = interoperability not achieved (from 0 to 15%), P = interoperability partially achieved (from 15 to 50%), L = interoperability largely achieved (from 51 to 85%) and F = interoperability fully achieved (from 86 to 100%) which demonstrate the fulfilment of the interoperability requirements (Figure 4). So, to evolve between the maturity stages, it is important to meet all the requirements. Achievement of capability indicators does not intend to be all-inclusive nor applicable in its entirety. It is possible to achieve different capability stages for different dimensions. For example, a system can achieve syntactic stage 5, semantic stage 4 and pragmatic stage 1.

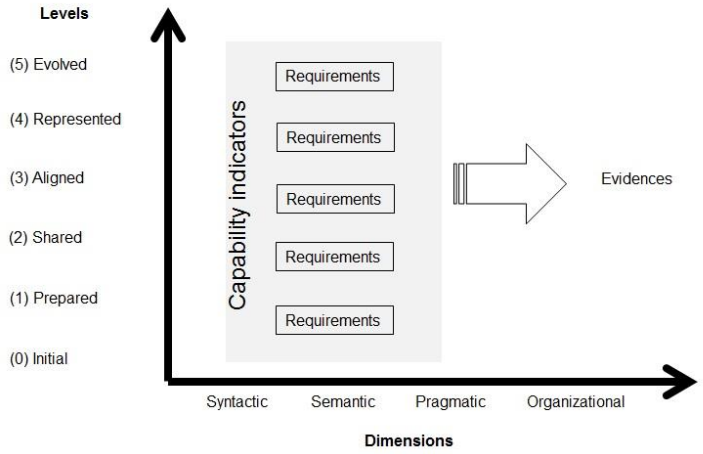

Fig. 3 Amortisse structure

The Capability indicators aggregate requirements used to assess system interoperability. These requirements are presented in the next section on tables 3, 4 and 5.

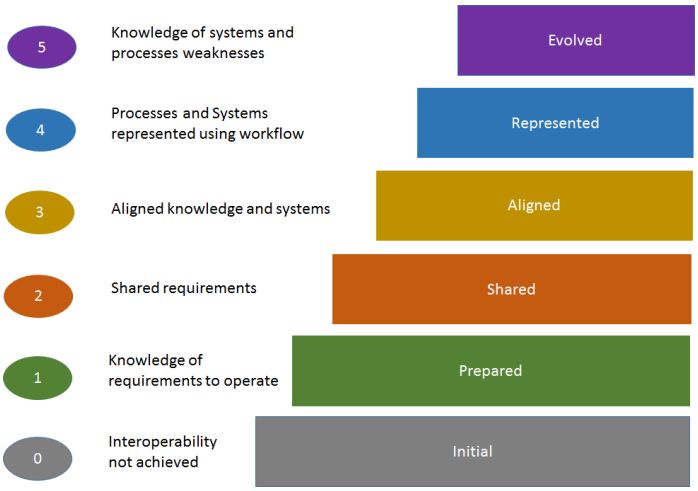

Fig. 4 Amortisse maturity stages

Figure 5 summarizes Amortise main elements and their relationships. Amortisse (Figure 5 (a)) contains interoperability requirements organized in dimensions, and levels. It is used to assess systems interoperability, i.e. communication, connection and collaboration among systems. This evaluation is performed by the involved stakeholders (Figure 5 (b)), which can be systems analysts, domain specialists, managers and so on. To perform the evaluation, it is necessary to analyze the involved systems, consider the domain requirements and collect data using techniques such as questionnaires, interviews or brainstorming (Figure 5 (c)). These are the input of the assessment process, which aims to generate as output the mature score for each dimension (Figure 5 (d)).

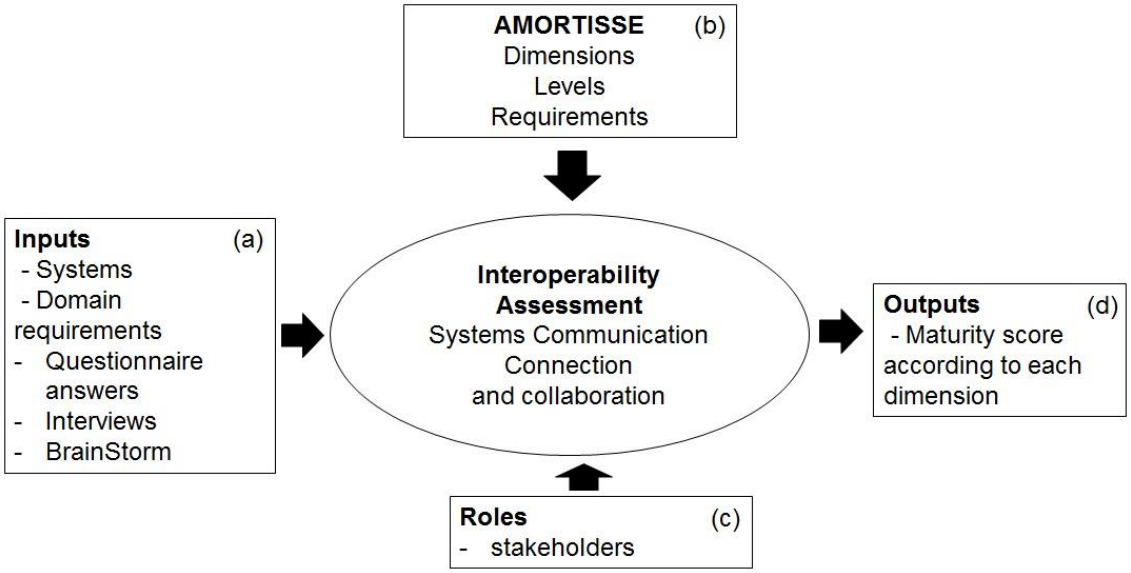

Fig. 5 Amortisse assessment process and its involved elements

4.1. Amortisse Specification

This section presents the Amortisse specification following the methodology life cycle presented in Figure 2. Frequently, maturity models lack a sound foundation or are derived based on an arbitrary design method. Amortisse was specified according to a pre-defined methodology to understand how each model element was specified and to support future extensions. The next subsections present the Amortisse specification according to each phase of the adopted methodology.

4.1.1. Scope Phase

Interoperability is usually classified in layers, representing potential barriers that hamper collaboration among systems. An interoperability architecture should combine network, platform, data, message, and software architecture to make systems communication possible. Table 2 presents the interoperability levels, i.e. the scope of our maturity model, explicitly ordered to give a view of the systems and technology. These levels and their components were selected and defined based on the most cited interoperability types pointed in literature reviews performed on (Ribeiro, 2019), (Santos, 2021). While syntactic and semantic interoperability concepts are well established, pragmatic and organizational concepts are not. However, current research on interoperability often points out that aspects related to message context (pragmatic level) and organization business (organization level) should be considered in systems collaboration (Ribeiro, 2019), (Santos, 2021). Therefore, the output of this phase was a component list (grouping) for the interoperability assessment. Table 2 presents the Scope phase output artifact, column 1 shows the interoperability levels, and column 2 its related components. For example, to meet the syntactic dimension, the network component must be considered, i.e. it would be difficult to exchange information between systems without considering network issues.

Tab. 2 List of components (groupings) generated as output of the Scope phase

Interoperability Level	Components
Organizational	Organizational and cooperation policies, workflows and business rules
Pragmatic	Use, intention, context, effect, meaning and understanding
Semantics	Language or taxonomy or ontologies
Syntactic	Devices (smartphones) Interface (browser) Data format (XML) and delivery (web services) Applications and protocols Database Framework Security Network

4.1.2 Design Phase

Following the proposed methodology, in the Design phase we should specify the MM architecture, i.e. define MM dimensions and stages.

Our model focuses on how maturity is measured, not on what maturity represents. So, we adopted the bottom-up approach to defining the maturity stages, i.e. we previously identified the interoperability components to then decide on interoperability levels. Based on the results of the Scope phase, the model architecture was structured into four dimensions, representing the levels of interoperability.

We also defined five maturity stages for each dimension, which represents a 5-point Likert scale. They are named Level 1, Level 2 and so on, expressing a logical progression. The Likert scale is often used to develop a maturity model (de Bruin, 2005). In our model, Level 0 represents a lack of capacity to interoperate; Level 1 represents certain knowledge about the requirements to interoperate; Level 2 represents shared knowledge of what needs to be done to interoperate; Level 3 represents an alignment of knowledge and systems to interoperate; level 4 indicates a representation of the processes and systems in workflows; and level 5 represents an understanding of weaknesses in systems and processes to evolve.

4.1.3 Model Content Phase

In this phase, it is necessary to find what needs to be measured in the maturity assessment model. So, the first task is to specify interoperability requirements for each component of each dimension. For our model, an initial requirements list (presented in tables 3, 4 and 5) has been created based on the components previously identified for the interoperability domain. For example, requirement ID 1 *Message exchanged in the interaction between systems services has the use intention for the receiver* is associated with the intention component in the pragmatic dimension and the requirement and requirement ID 11 *The policies and specifications of the base applications and protocols for system operation are understood* is related to the syntactic interoperability protocol component (see Table 5).

Then, the eligibility of the requirements must be assessed, and as a consequence, requirements can be added or removed. We chose a pilot study to determine the requirements eligibility when participants should point out if the requirement should comprise the model content list or not.

The study context was a widely used scenario, the interoperability between a hospital and a laboratory information system (Asuncion, 2011) concerning two situations: the lab cannot understand the intent of the hospital because the lab has a different understanding of the context in which the hospital is operating (Figure 6.a); the laboratory can understand the intention of the hospital. Now both have the same understanding of context (Figure 6.b). For example, considering that the hospital is in an emergency context and requests a report from the laboratory. If the laboratory understands this request as usual, it will respond to it as a non-urgent routine request (Figure 6.a). On the other hand, if the laboratory can perceive the hospital's intention, the response will be sent urgently (Figure 6.b). To support the study we developed a web software according to this scenario. The software considers that laboratories offer services, among them the report service in level of urgency, emergency and standard. So, the service consumer, e.g. the hospital, sends contextual information to characterize the request. This contextual information is represented through schemas containing the context dimensions, for example, how the data will be obtained, who is the requestor, what the entity is doing or intends to do and why the action. A web service checks whether the message has the information needed to perform queries based on the intent and the contextual information and search the service in a base. If so, the WSDL link is obtained and sent to the requester to make the service call. Otherwise, the requester is notified that the information is not sufficient to obtain the service.

We conducted a pilot study with seven academic participants, doctors and master students working on an interoperability research project. For each requirement (tables 3, 4 and 5), participants evaluated their eligibility by answering a questionnaire through semi-structured interviews. The questionnaire was structured following the four interoperability dimensions (syntactic, semantic, pragmatic, and organizational), where each question represents a requirement (from tables) of one of these dimensions. For instance, the question "*In this scenario, do messages exchanged in interaction among systems services contain the use intention for the receiver?"* refers to requirement ID 1 (Table 3). Likewise, thirty-one questions were specified, all of them using a 5-point Likert scale which is assumed to have different levels of difficulty. So, participants indicated the difficulty of achieving its requirement in the scenario, five for extremely difficult to one for extremely easy. Based on the collected answers, the list of the initial requirements was then measured by basically counting answered questions.

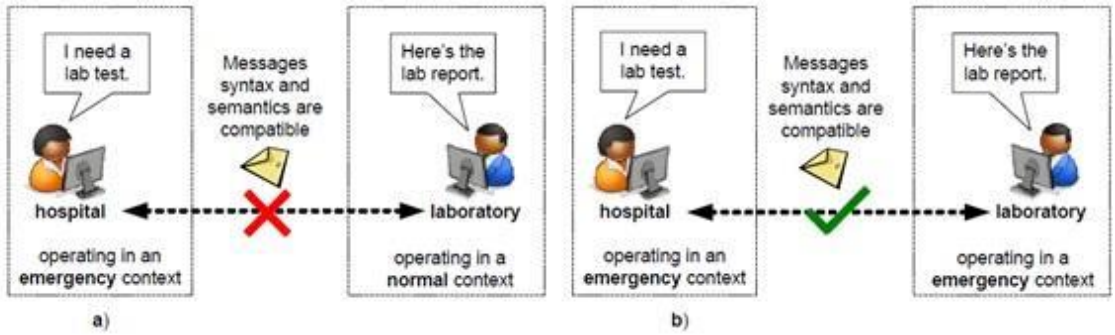

Fig. 6 (non-)pragmatically interoperable scenarios (Asuncion, 2011)

Tab. 3 Requirements of Pragmatic and Organizational dimension

ID	Pragmatic Interoperability Requirements
1	Message exchanged in interaction among systems services contains the use intention of the receiver
2	Message intention is clearly and explicitly defined
3	Message is sent automatically, via a communication channel established between parties
4	A standard to represent message intention is unambiguously established
5	A standard to represent message intention is understood and shared by involved parties in interaction
6	Message effect is compatible with your intention
7	The exchanged message context is clearly and explicitly defined
8	A context standard of the exchanged message is unambiguously established
9	The context standard of the exchange message is understood and shared

Tab. 4 Requirements of Organizational dimension

ID	**Organizational Interoperability**
10	Organizational intentions, such as business rules and organizational policies are clearly and explicitly defined
11	An organizational intent standard is unambiguously established
12	Organizations intention pattern is understood and shared
13	A continuous communication channel among the parties is established
14	Responsibilities and agreements are established among the parties
15	Responsibilities and agreements are performed, negotiated and monitored
16	Information about service use (who, how, where, when) is stored and shared
17	Collaborators' context (social, cultural, workspace and so on) is clearly and explicitly defined
18	A standard to represent the collaborators' context is unambiguously established
19	A standard for representing collaborators' context is understood and shared by the involved parties in collaboration
20	Mechanisms to support the trust among the parties are established
21	Organizational intentions are enriched with historical information about shared resources and collaborators' contexts, to better meet the expectations

Tab. 5 Requirements of Semantic and Syntactic dimension

ID	**Semantic Interoperability**
22	Among systems, disjoint concepts are represented unambiguously
23	There are strategies to represent, unambiguously, general concepts
24	There are strategies to represent, unambiguously, specific concepts
25	There are strategies to represent, unambiguously, concepts with overlapping meanings used in interactions among systems
26	Among systems, there is no ambiguity regarding the concepts (general, specific, overlapping)
	Syntactic Interoperability
27	Network infrastructure policies and specifications are understood
28	Applications policies and specifications and base protocols for system operation are understood
29	Policies and specifications that define a uniform way to the transit of encoded data are understood
30	Policies and specifications that define the rules for encoding data using the representation of structures and descriptors according to standardized languages are understood
31	Policies and specifications to provide access to all the information systems are understood

4.1.4 Model Construction Phase

The main goal of this phase is to generate the initial maturity model according to the elements specified in the previews phase: dimensions, stages and requirements. So, the requirements just elected, based on the degree of difficulty in their achievement, must be distributed in the stages of their respective dimension.

Each requirement difficulty, defined in the Model Content phase according to the scenario adopted, i.e. the questionnaire answers, were inserted for analysis in the Rasch algorithm using the Ministep Software (Liu, 2010), i.e. using the Rasch algorithm, capabilities and difficulties are tested against estimates based on an item response function. So, difficult requirements have a higher measure than easy ones (Liu, 2010). The result of this step is a flat list of ordered requirements, comprised of columns one and two of Table 6. In column one, requirements are represented in the format of the Rasch algorithm. Therefore, $Coef$i08 represents the requirement of ID 8, semantic dimension, *Systems use the same symbol to represent the disjoint and overlapping concepts* which achieve the measurement 15 343 (Table 6, column 2).

Although empirical conclusions can be drawn from such a list, the item difficulty measures do not allow the requirements among the maturity stages. For instance, should requirements 08 and 02, which reached item difficulty values 15.343 and 14.815, be placed in the distinct or same stage? Therefore, to carry out this allocation, once the results were extracted, it was necessary to put values in a cluster analysis to get the Amortisse draft version finally. The cluster analysis is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense) to each other than to those in other groups (clusters) (Romesburg, 2004). The clustering algorithm grouped the difficulty items into stages. That is, the analysis of the values achieved by each requirement difficulty item indicates at what dimension stage of maturity the requirement should be allocated in the MM. For clustering, we used

hierarchical cluster analysis of SPSS (squared Euclidean distance, Ward's method (Field, 2013)) and set the desired number of maturity stages to five, Table 6 column three. Table 7 presents a slice of the Rasch results organized according to Amortisse levels and dimensions, and Table 8 presents the same data using the requirements description.

Tab. 6 Rasch algorithm and Cluster analysis: results slice

Requirements	Item difficulty	Level / Stage
$Coef$i08	15.343	5
$Coef$i02	14.815	5
$Coef$i04	13.107	4
$Coef$i20	13.107	4
$Coef$i29	2.435	3
$Coef$i03	1.762	2
$Coef$i05	0.905	1
$Coef$i07	0.519	1

Tab. 7 Maturity Model Draft Version

Dimension	Level 1	Level 2	Level 3	Level 4	Level 5
Organizational	$Coef$i18	$Coef$i14	$Coef$i21	$Coef$i20	$Coef$i10
Pragmatic	$Coef$i09	$Coef$i03		$Coef$i04	$Coef$i02
Semantics	$Coef$i26		$Coef$i23	$Coef$i24	$Coef$i22
Syntactic	$Coef$i27	$Coef$i31	$Coef$i29		$Coef$i30

Tab. 8 Maturity Model slice.

	Level 1	Level 2	Level 3	Level 4	Level 5
Organizational	A standard to represent the collaborators' context is unambiguously established	Responsibilities and agreements are established among the parties	Organizational intentions are enriched with historical information about shared resources and the collaborators contexts, to better meet the expectations	Mechanisms to support the trust among the parties are established	Organizational intention, such as business rules, and organizational policies are clearly and explicitly defined
Pragmatic	The context standard of the exchange message is understood and shared	Messages are sent automatically, via a communication channel established between parties		A standard to represent message intention is unambiguously established	Message intention is clearly and explicitly defined
Semantics	Among systems, there is no ambiguity regarding the concepts (general, specific, overlapping)		There are strategies to represent, unambiguously, general concepts	here are strategies to represent, unambiguously, specific concepts	Among systems, disjoint concepts are represented unambiguously
Syntactic	Network infrastructure policies and specifications and understood	Policies and specifications to provide access to all the information systems are understood	Policies and specifications that define a uniform way to the transit of encoded data are understood		Policies and specifications that define the rules for encoding data using the representation of structures and descriptors according to standardized languages are understood

4.1.5 Implementation and Maintenance Phase

After obtaining an initial version of our interoperability maturity model, we should apply it in a real scenario to verify its applicability. Therefore, we performed a case study, which description is detailed in Section 5.1. The Maintenance phase is represented in the Exploratory study, section 5.2.

5. Amortisse Assessment

Validations, which involve humans, are very challenging since they usually have many activities that people with different levels of expertise perform. Therefore, they should be carried out in phases, where each phase is an evolution from the previous one. Amortisse evaluation was carried out in three phases: (i) a case study to evaluate Amortisse applicability in the web and internet domain; (ii) a case study to evaluate Amortisse evolution using a cloud domain; and (iii) an interview to get the user perception about aspects related to Amortisse usefulness, consistency, scope, among others. So, the first case study has shown Amortisse feasibility, the second case study has shown Amortisse evolution to another domain and the interview the user perception of different aspects when using Amortisse. Replications of these studies are being planned.

We essentially adopted qualitative methods due to the limited number of people with knowledge in both the interoperability domain and maturity evaluation in organizations.

5.1. Applicability Case Study

To evaluate the applicability of using Amortisse in real scenarios, we performed a case study on an e-government application hosted on the web platform. The study adopted Amortisse to identify the interoperability maturity level of a set of systems of a governance organization in Brazil that deal with systems for traffic control. It was performed at a government partner software house, developing services for citizens and several government organizations. The study was conducted according to the guidelines proposed by Wohlin et. al. (Wohlin, 2012).

5.1.1. Case Study Design

The study identifies the interoperability maturity level of three software systems: a mobile application that allows traffic guards to apply traffic fines, a web portal for citizens to query their driver's license information, and an information system (desktop client system) to manage enforcement penalties laws on drivers. These systems allow for a collaborative relationship among the city hall administration, state government and the population regarding traffic public order and tickets.

The main goal of the study was defined according to Goal Question Metric (GQM) (Basili, 1994) template in Figure 7. Based on this goal, we defined two questions to guide the validation.

Question 1: Do Amortisse MM dimensions cover the interoperability scope needed to be measured? With this question, we want to identify if the dimensions were well defined, i.e. whether Amortisse comprises the necessary dimensions.

Question 2: Are Interoperability requirements sufficient to measure interoperability in every dimension? With this question, we want to identify if the requirements are sufficient to measure maturity in each dimension and level.

Analyze Amortisse **For the purpose of** evaluating its feasibility in measure the interoperability level of systems **Related to** dimensions, levels and requirements **From the respective of** system analysts **In the context of** a set of information systems in a real company

Fig. 7 Case study main goal according to GQM template

Participants are company employees with knowledge of software infrastructure and development. We selected seven participants by applying a questionnaire to assess their skills and knowledge about interoperability. Invitations were sent by email by the company manager. There is one Senior software architect, two full network analysts, two full software engineers, one Junior analyst, and one DevOPS. Most of them (71%) are over 30 years old, and 57% have worked in this company for more than four years.

Therefore, they have proper knowledge of the organizations' business process and technological infrastructure.

Participants applied Amortisse MM to measure the interoperability level of the involved systems. Data collection was carried out directly by applying an interview guided by the researcher through an online questionnaire during the study execution. So, the participants should answer, for each requirement of tables 3, 4 and 5, if it was met or not by the analyzed systems.

5.1.2. Study Execution

The study lasted two and a half hours and was conducted in three phases: (i) initially, the researcher introduced Amortisse to the participants; (ii) then, participants analyzed which Amortisse requirements were met by the systems. In order to do this, they had to collect some evidence (e.g. identify features, data structure, data conversion routines, among others) to guide their answers about each requirement fulfilment, using a Linkert scale (e.g. 5 - the requirement is completely met in the systems, 1 - the requirement is not met.). This analysis was guided by an online questionnaire; (iii) finally, participants evaluated in a brainstorming session the requirements disposition through model dimensions and levels.

During the study, the researcher played the role of presenter in the first phase and interviewer in the other two phases.

5.1.3. Data Analysis

The collected data can be analyzed using different methods, e.g. statistical and qualitative methods. Due to the number of participants, we decided to adopt a qualitative one, a brainstorming session. Therefore, we promoted a group discussion where participants analyzed all the questionnaire data for each question to achieve a consensus.

In summary, the answers related to syntactic interoperability showed a disagreement among participants, while in the other dimensions, the results were balanced. This disagreement can be explained due to the variation of knowledge among the participants. For example, junior analysts and full software engineers are closer to the organization's business and functional requirements, while network analysts and DevOps are involved with technology infrastructure. As a highlight, it was essential to have a software architect profile as a group member, and he had a holistic view of the software systems.

After the brainstorming session, participants updated their answers and achieved a consensus. Participants' responses were summarized in tables 9, 10, 11 and 12 according to each dimension. The (+) markings indicate that the system fulfils the requirement, i.e. more than 75% of the responses state that the system's requirements were 4 (met) or 5 (wholly met), and the (-) markings indicate that the system did not achieve the requirement.

Tab. 9 Organizational Interoperability

Levels	Organizational Interoperability Dimension
1	(+) Collaborators' context (social, cultural, workspace and so on) is clearly and explicitly defined (+) Responsibilities and agreements are performed, negotiated and monitored (-) An organizational intent standard is unambiguously established (-) A continuous communication channel among the parties is established (+) A standard to represent the collaborator´s context is unambiguously established
2	(+) Responsibilities and agreements are established among parties (+) A standard for representing collaborators' context is understood and shared by the involved parties in collaboration
3	(+)organizational intentions are enriched with historical information about shared resources and collaborators' contexts, to better meet the expectations
4	(-) An organizations' intention pattern is understood and shared (+) Mechanisms to support the trust among the parties are established (+) Information about service use (who, how, where, when) is stored and shared
5	(+) Organizational intentions, such as business rules and organizational policies are clearly and explicitly defined

Tab. 10 Pragmatic Interoperability

Level	Pragmatic Interoperability Dimension
1	(-) A standard to represent message intention is understood and shared by involved parties in an interaction (-) The message exchanged in interaction among systems/services contains the use intention of the receiver (+) The context standard of the exchange message is understood and shared
2	(+) A standard to represent the context in which the exchanged message is inserted is unambiguously established (+) The message is sent automatically, via a communication channel established between parties (+) The exchanged message context is clearly and explicitly defined
3	
4	(-) A standard to represent message intention is unambiguously established
5	(+) The message effect is compatible with your intention (-) Message intention is clearly and explicitly defined

Tab. 11 Semantic Interoperability

Level	Semantic Interoperability Dimension
1	(+) There are strategies to represent, unambiguously, concepts with overlapping meanings used in interactions among systems (+) Among systems, there is no ambiguity regarding the concepts (general, specific, overlapping)
2	
3	(+) There are strategies to represent, unambiguously, general concepts
4	(+) There are strategies to represent, unambiguously, specific concepts
5	(+) Among systems, disjoint concepts are represented unambiguously

Tab. 12 Syntactic Interoperability

Level	Syntactic Interoperability Dimension
1	(+) Network infrastructure policies and specifications are understood
2	(+) Applications policies and specifications and base protocols for system operation are understood (+) Policies and specifications to provide access to all the information systems are understood
3	(+) Policies and specifications that define a uniform way to the transit of encoded data are understood
4	
5	(+) Policies and specifications that define the rules for encoding data using the representation of structures and descriptors according to standardized languages are understood

Amortisse achieves complete adherence on stages 2 and 3 for all interoperability dimensions according to these data. Four requirements must be considered at stage 1, two of them at the stage of pragmatic interoperability and two at the organizational stage. At stage 4, two requirements were not fulfilled, in the organization and pragmatic dimensions. Finally, at stage 5, one requirement is missing at pragmatic interoperability.

Table 13 summarizes the systems scores in this case study according to each dimension and level. As can be seen, there is no pre-requirement among the levels, i.e. it is possible, for example in the organizational dimension, to achieve 60% at level 1 and 100% at level 2.

Tab. 13 Scores Summary

Dimension/ Level	1 Prepared	2 Shared	3 Aligned	4 Represented	5 Evolved
Organizational	**60%**	**100%**	**100%**	**66%**	**100%**
Pragmatic	**30%**	**100%**	**-**	**0%**	**50%**
Semantic	**100%**	**-**	**100%**	**100%**	**100%**
Syntactic	**100%**	**100%**	**100%**	**-**	**100%**

Table 14 shows the possible scores of each stage, i.e. which requirements must be met to achieve each level, considering a vertical progression between dimensions. For example, to achieve level 1, it is necessary to meet eleven requirements: five of organizational interoperability, three from pragmatics, two from semantics, and one from syntactic. As we can see in the score achieved in this case study, the company reached 7 of the 11 possible requirements. If the company obtains 100% at all levels, it has achieved full interoperability. In the horizontal progression, i.e. into the dimensions, we realized that the organization obtained 100% in the syntactic and semantic dimensions, which can ensure complete connectivity between systems and information correctly interpreted by the receiving system.

Tab. 14 Possible and achieved score in each level

	Level 1	Level 2	Level 3	Level 4	Level 5
Possible score	11	7	3	5	5
Achieved score	7	7	3	3	4

Related to Question 1, *Do tAmortisse MM dimensions cover the interoperability scope needed to be measured?* participants highlighted the importance of representing dimensions as interoperability types since there is a relationship between them and could realize the relationship among them. For instance, syntactic interoperability is necessary to achieve semantics and so on. Therefore, we have a positive evaluation of Question 1, i.e. Amortisse dimensions fulfilled the company's needs. Concerning the requirements identified for each dimension (*Question 2: Are Interoperability requirements sufficient to measure interoperability in every dimension?*), Amortisse is composed of thirty-one interoperability requirements, which are an initial effort to verify the interoperability of software systems in different domains. Participants considered this set of requirements enough to verify the interoperability of the company scenario. During the study, execution participants did not point out unnecessary requirements or missing ones. These requirements do not intend to be exhaustive nor apply in their entirety. Suitable subsets to the context and scope of the assessment should be selected and possibly increased with new requirements. Therefore, Question 1 and Question 2 evidence the Amortisse applicability to measure the interoperability maturity of these systems. Additionally, we could observe several interoperability requirements were implemented within the application source code and not by independent services as in current academic research trends, i.e. using traditional software development methods, where functional and non-functional requirements are identified in early systems development stages. Therefore, new requirements could be added to Amortisse level five to measure this feature.

The results presented in tables 9, 10, 11, and 12 allowed us to identify the strengths and weaknesses within the organization, supporting investment according to its real needs. It is up to the organization to understand and decide the areas that still need more attention and investment in line with current capabilities to interoperate.

5.1.4. Threats to validity

To prevent bias in the validation, we now discuss some threats to its validity. Regarding internal validity, to decrease the threats regarding the participants' level of expertise, the study selected specialists and applied a questionnaire to ensure their knowledge. The study was performed in a company with a limited number of participants concerning external validity. So, results generalization is limited to scenarios that share similar characteristics. Related to construct validity, we followed Wohlin (Wohlin, 2012) guidelines to perform the case study activities. Finally, concerning conclusion validity, the study did not use a statistical method to evaluate the results, which may influence the conclusion. Besides this, concerning the methodology proposed to guide Amortisse's specification, its use in new domains may require adaptations of the defined activities and artifacts.

5.2. Evolution Case Study

This study aims to assess Amortisse's ability to evolve to be used in new domains. We choose a cloud computing scenario, which presents several distinct interoperability issues from the web applications scenario used as a reference to specify Amortisse's initial version. Therefore, the Maintenance phase suggested guidelines were followed (Section 4.1.5). So, while the Amortisse case study presented in Section 5.1 considered a scenario with a traditional web architecture based on three independent systems, now we

adopt a scenario with Service Oriented Architecture (SOA) that relies on both cloud and internet environments. In this way, the results of this study should identify which requirements need to be excluded or added for Amortisse to be able to measure interoperability issues in this new scenario.

5.2.1. Case Study Design

The study's main goal is to analyze Amortisse's ability to evolve into new domains. This goal is defined according to GQM template (Basili, 1994) in Figure 8. Based on this goal, we defined the following research question to guide the study:

Question: Can Amortisse evolve to analyse interoperability in a different domain from its first version?

With this question, we aim to observe if it is possible to customize the requirements previously defined for the Amortisse model, i.e. include, exclude or change some requirements.

Analyze Amortisse **For the purpose of** evaluating its capacity for evolution **Related to** requirements customization **From the perspective of** systems analysts **In the context of** new scenarios

Fig. 8 Evolution study goal according to GQM template

This case study was applied to a software company that offers a Dashboard infrastructure. The Dashboards can be customized for different purposes in which images and text information can be displayed. For instance, they have dashboard clients for public or private surveillance needs. This infrastructure comprises several services (e.g. Keystone, Image, Networking, DataStore) that should use different APIs (Table 15). Currently, the software company offers thirteen different services, and Clients may request some specific cloud platforms to be used.

Tab. 15 Services of Dashboard cloud infrastructure

Service name	Environment name	Service description
Identity Service	Keystone	User management
Compute service	Nova	Virtual machine management
Image service	Glance	Manages Virtual image distribution
Dashboard	Horizon	Provides GUI console via a Web browser
Object storage	Swift	Provides Cloud Storage
Block storage	Cinder	Storage Management for Virtual Machine
Network service	Neutron	Virtual Networking Management
Orchestration Service	Heat	Provides Orchestration function for Virtual Machine
Metering Service	Cailometer	Provides the function of Usage measurement for accounting
Database Service	Trove	Database resource Management
Data Processing Service	Sahara	Provides Data Processing function
Bare Metal Provisioning	Ironic	Provides Bare Metal Provisioning function
Messaging Service	Zaqar	Provides Messaging Service function
Shared File System	Manila	Provides File Sharing Service
DNS Service	Designate	Provides DNS Server Service
Key Manager Service	Barbican	Provides Key Management Service

Six employees participated in this study, one was a system analyst, one was a networking expert, and four were software developers experts on cloud platforms. Most of them (71%) are over 30 years old. At least half have more than four years of experience working on cloud platforms and have participated in the entire development life cycle, now working on its maintenance and customization for new customers. As stated before, the scenario comprises a set of services available to the end-user through a dashboard. Final users can interact with systems through a dashboard. All services are authenticated through a typical identity service, and services interact through public APIs (Application Program Interfaces), except when system administrator commands are required, CLIs are used.

The study was conducted in phases: (i) initially, participants measured interoperability in the new scenario using Amortisse; (ii) at the end of this activity, they have to identify requirements that do not fit the cloud computing scenario as well as proposed a list of new requirements specific for the new scenario that should be included in Amortise; (i) finally, the participants indicated the dimension and level which the new requirement should be placed according to Amortise structure.

5.2.2. Study Execution

The study lasted three hours. First, the participants received training, including introducing the Amortisse model. After that, to measure the infrastructure interoperability in the new scenario using Amortisse, we repeat the same strategy of the case study. The participants filled out a questionnaire answering about each requirement. All questions used a Likert scale ranging from one to five (one - less effort, five - more effort). In summary, the infrastructure only fulfils the syntactic interoperability requirements. Second, in a brainstorming section, participants discussed whether cloud requirements would be eligible and if some should be excluded for Amortise. Table 16 presents the cloud requirements suggested by the participants. At this point, consensus about the requirements eligibility was considered. Participants stated that no requirement should be removed from Amortisse because they are planning additional functionalities and these requirements should be useful. Finally, they answered a questionnaire to determine each requirement dimension and level. All questions used a Likert scale ranging from one to five (one - less effort, five - more effort).

Tab. 16 Requirements Questionnaire

Requirement	**Description**	**Eligibility (Yes/No)**	**Effort (1 to 5)**
Cost	Cost can influence cloud type.		
Time-to-market	The ability to deliver products or services within a flexible time frame is a common factor in creating a cloud.		
Revenue opportunity	Revenue opportunities vary by cloud intent and use case.		
Compliance and geo-location	Organizations may have legal obligations and regulatory compliance measures that may require a certain workload or data that is not located in certain regions.		
Security	The importance of security varies depending on the type of organization that uses a cloud. For example, financial institutions generally require higher security requirements.		
Service level agreement	Service level agreements (SLAs) must be developed in conjunction with business, technical and legal information.		
Migration, availability, site loss and recovery	Organizations using cloud-based services can embrace business diversity and adopt a hybrid cloud design to spread their workloads across multiple cloud providers.		
Network design	Includes decisions on the need for interconnection.		
Support and maintenance	It considers maintenance, support and availability tasks.		
SLA considerations (operational)	Define availability levels that will impact the cloud design cloud to provide redundancy and high availability.		
Logging and monitoring	Clouds require appropriate monitoring platforms to identify and manage errors.		
Data plane and control plane	Maintain the availability of instances, networks, storage, and additional services.		
Site loss and recovery	Strategies must be implemented to understand and plan for recovery scenarios.		

1.1.1. Data Analysis

The participants' requirements analysis indicated that only the time-to-market requirement should not be placed for this new scenario, perhaps because it is a requirement with high granularity and is difficult to measure. Unlike the three systems information used in the first case study, this dashboard is a basic infrastructure that may be customized for different purposes. Thus, beyond Cloud Platforms aspects, this specificity impacted the participants' analysis.

Related to the case study question, we can answer affirmatively. The participants could perform all the proposed steps to update Amortisse for Cloud Domains and then apply the Amortisse new version to measure the Dashboard Infrastructure.

1.1.2. Threats to Validity

The threats to the validity of this study are similar to the ones presented in the case study. However, it is important to highlight the possible weakness in the method used to obtain the cloud requirements, which the study participants suggested. To decrease the uncertainty, we applied the survey to specialists with more than four years of experience.

1.2. Interview Study: User Perception Evaluation

To get insights into the user perception when using Amortisse, at the end of the case studies, participants were invited to a semi-structured interview. So, twelve participants were interviewed and gave their opinion concerning the following Amortisse aspects *usefulness*, *consistency*, and *clarity*.

The interview was organized into three questions, one for each aspect investigated, as follows.

Question 1: What is Amortisse usefulness as a supporting tool for interoperability assessment? In this question, we want to understand how much Amortise facilitated or hindered systems interoperability assessment.

Question 2: How do you rate the consistency among the requirements, dimensions and levels used in the Amortisse model? With this question, we want to identify whether the participants understand and agree with the relationships among these elements.

Question 3: How easy is it to apply Amortisse, i.e. is it clear how the systems evaluation process should be carried out? In this question, we want to identify whether the instructions for using Amortisse can be understood by participants.

The interview was conducted by the researcher. Participants sent their answers through an electronic form using the following scale: 1 - very bad, 2 - bad, 3 - neutral, 4 - good, and 5 - very good.

The results of this questionnaire were analyzed according to each question. Regarding *Usefulness*, 66.6% of the participants rated Amortise as *very good* and 33.3% as *good* in evaluating systems interoperability evidencing that Amortisse is helpful and facilitates the assessment process.

Related to *Consistency* and *Clarity*, for both, all participants rated as *very good* (50%) or *good* (50%). This indicates that for the participants the model elements, i.e. dimensions, levels and requirements, are clearly defined and coherently related.

Table 17 summarizes some participants' comments during the interview.

Tab. 17 Participants' comments during the interview

Aspect	Participant comment
Utility	"Amortisse makes us think about the solutions adopted and plan for future ones."
Clarity	"I believe that the most critical point is the requirements description. It could be improved."
Clarity	"The strategy seems to be comprehensive for the cloud context. However, requirements need a more detailed description to improve evidence collection."
Consistency	"Amortisse visual diagramming and modelling make interoperability concepts easy to understand."

These are initial data with important feedback that will be carefully analyzed to improve the model. In summary, the interview results indicate a positive perception of the participants regarding the Amortisse use as a tool to assess systems interoperability. However, it is important to highlight that these assessments were carried out in the researcher's presence. Therefore, we conclude that it will be necessary to improve the requirements description and develop new artefacts to guide the MM application and customization.

This initial study has a potential bias related to the small size of the participant. So, as we know that it is essential to replicate this interview every time the case study is performed. Another primary source of uncertainty is the method used to obtain the evaluated aspects, which was through the author's expertise.

1.3. Discussion

Amortisse is a maturity model to support systems interoperability assessment in software systems. Additionally, Amortisse's rationale is systematized in a methodology, which enables its evolution. e.g. requirements may be included or excluded to fit different scenarios. The methodology does not intend to be complete, but a reference both for Amortisse's specialization and for assisting the specifications of new models. Continuous improvements can be applied when new elements are needed. Instead of high-level requirements that usually comprises other interoperability assessment models, Amortisse is suitable for the organization needs that guided its first version. For instance, requirements of the syntactic level explicitly cope with web technologies aspects, as it is the current software development platform. Additionally, as the organization has vehicle traffic issues, context information should be considered in their systems collaboration. So the pragmatic interoperability was included as an Amorisse level. Interoperability MM usually does not consider pragmatic interoperability type as part of its structure (Rezaei, 2014).

Amortisse's first version was specified based on a literature review. We could consider a base format, which shows the model elements and how they should be used in the assessment process. The specification methodology may be perceived as an explanatory format, which shows step-by-step how the model was specified according to the defined methodology. This strategic goal was to enable the adoption of the MM and make it easy to define new elements to compose the model, i.e. the knowledge specified for Amortisse is available to be updated, following the methodology, towards its customization of new scenarios.

Amortisse's validation process has been incremental, and essentially qualitative. The quantitative method requires an adequate number of study participants. Literature and industry show that this kind of assessment better uses few specialists than a large number of participants with common Knowledge. Therefore, we carried out two studies that assess the MM applicability and its ability to evolve and evaluated cross-cutting aspects, such as usefulness and consistency, according to user perception. Although these three validations do not provide statistical analyses, they were conducted with experts and showed evidence for the proposal's feasibility. During the evaluation process, we noticed that the industry professionals have some knowledge about interoperability as a non-functional requirement but are not aware of all its categorization levels. During Amortisse's explanation, some questions arose, but often the professionals related some previous situations faced while developing a system. As a highlight, standards and middleware platforms were often the software solution pointed to achieve technical and syntactic interoperability. However, the inside system code solution was often applied to achieve other higher levels of interoperability. In that case, interoperability is developed using low cohesion and high coupling solutions. Amortisse usage could help offer an opportunity to think about new possibilities and, as an insight, high cohesion and loosely coupled interoperability solutions should be considered as new requirements.

Although the web domain has solved technical, syntactic, and semantic interoperability issues, Cloud, IoT, and Smart domains present new ones. We considered, therefore, that Amortisse offers a new strategy compared to the current models since it describes the model construction enabling it to be evolved. This strategy is especially important in a constantly evolving domain such as interoperability.

2. Conclusion and Future Work

Maturity models aim to assist organizations in improving the way it performs, a certain aspect guiding them towards the desired achievement. This paper introduced the Amortisse, a maturity model for software and system interoperability assessment, and a MM methodology, which makes its rationale explicit. It has been evaluated gradually through different methods.

Initially, a methodology for MM specification was proposed. Then, based a well known academic scenario, Amortisse first version was specified. After that, Amortisse was evaluated in a case study and in an exploratory study applied to different domains. The results of these studies provided evidence that in these scenarios, Amortisse was feasible to measure the interoperability of the involved systems, i.e. most of the participants agreed that it can be used to assist measure interoperability.

Adopting a methodology to conduct Amortisse specification makes the Amortisse extension easier for other domains. The methodology provides step-by-step the activities that should be performed to define new requirements, distribute them into the dimensions and levels and so on, i.e. all the rationale involved in the model specification is documented for future adaptations. This rationale is also essential to provide data compared to other maturity models.

The results of this paper show our strategy feasibility as Amortisse was able to measure the interoperability maturity of the systems, and the methodology allows the development of the first version and its maturity model evolution, i.e. the customization. The value to organizations of applying such a model lies in measuring and assessing domain capabilities at a given point in time. Amortisse can help organizations better understand existing domain capabilities, enabling benchmarking among other organizations. It can be an important instrument to support their interoperability goals, and contributing standardization.

For future work, Amortisse should be updated using evaluation results. To improve both the Amortisse and the specification methodology, we are now working on the case study replication in other domains. Besides, statistical methods are being considered to be used in data of different case studies, aiming to give a better understanding of the case study results. By sharing our proposal, we aim that the community could perform new replications and, therefore, bring improvements and insights to the Amortisse proposal.

Acknowledgements

This study was supported by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) - Grant number 1555173 and Finance Code 001.

Conflict of Interest Statement

I declare that the authors have no competing interests as defined by Springer, or other interests that might be perceived to influence the results and/or discussion reported in this paper.

References

Almeida, Sandra Kawamoto and Jorge Rady de. 2017. Scrum-DR: An extension of the scrum framework adherent to the capability maturity model using design rationale techniques. *CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON).* 2017, pp. 1-7.

Asuncion, C. H. 2011. Pragmatic interoperability in the enterprise - A research agenda. *Theoretical Computer Science - TCS.* 01, 2011, Vol. 731.

Basili, Victor R. and Caldiera, Gianluigi and Rombach, H. Dieter. 1994. The Goal Question Metric Approach. *Encyclopedia of Software Engineering.* s.l. : Wiley, 1994.

Becker, Jörg and Knackstedt, Ralf and Pöppelbuß, Jens. 2009. Developing Maturity Models for IT Management. *Business & Information Systems Engineering: The International Journal of WIRTSCHAFTSINFORMATIK.* 6 de 2009, Vol. 1, 3, pp. 213-222.

Boscarioli, Clodis and Araújo, Renata Mendes and Maciel, Rita Suzana Pitangueira. 2017. *I GranDSI-BR--Grand Research Challenges in Information Systems in Brazil 2016-2026.* 2017. 978-85-7669-384-0.

Braga, R. S. P. Maciel and J. M. N. David and D. B. Claro and R. 2017. Full Interoperability: Challenges and Opportunities for Future Information Systems. *Grand Research Challenges in Information.* s.l. : SBC, 2017, pp. 107-118.

Brocke, Jan vom. 2019. Design Principles for Reference Modelling: Reusing Information Models by Means of Aggregation, Specialisation, Instantiation and Analogy. s.l. : European Research Center for Information Systems, 2019.

Carvalho João Vidal and Rocha, Alvaro and Abreu, Antônio and Afonso, Ana. 2017. Development methodology of the HISMM Maturity Model. [ed.] IEEE. *2017 12th Iberian Conference on Information Systems and Technologies (CISTI).* 2017.

Carvalho, João Vidal and Rocha, Álvaro and van de Wetering, Rogier and Abreu, Antônio. 2019. A Maturity model for hospital information systems. *Journal of Business Research.* 2019, Elsevier.

Chen, David. 2013. Framework for enterprise interoperability and maturity model (CEN/ISO 11354). *Interoperability for enterprise software and applications.* 2013, pp. 15-22.

Chrissis, Mary Beth and Konrad, Mike and Shrum, Sandra. 2011. *CMMI for Development: Guidelines for Process Integration and Product Improvement.* 3. s.l. : Addison-Wesley Professional, 2011. 0321711505.

Clark, Thea and Jones, Richard. 1999. Organisational interoperability maturity model for C2. [ed.] Citeseer. *Proceedings of the 1999 Command and Control Research and Technology Symposium.* 1999, Vol. 29.

Correa. 2012. Metodologia para aferição do nível de maturidade associado à interoperabilidade técnica nas áreas de Governo Eletrônico. *CEATEC - Centro de Ciências Exatas, Ambientais e de Tecnologias.* 2012.

Cosic, Ranko and Shanks, Graeme and Maynard, Sean. 2012. Towards a business analytics capability maturity model. *Proceedings of the 23rd Australasian Conference on Information Systems 2012.* s.l. : ACIS, 2012.

De Ayala, RJ. 2018. Item response theory and Rasch modeling. *The reviewer's guide to quantitative methods in the social sciences.* s.l. : Routledge, 2018, pp. 145-163.

de Bruin, Tonia and Freeze, Ronald and Kulkarni, Uday and Rosemann, Michael. 2005. Understanding the Main Phases of Developing a Maturity Assessment Model. *Australasian Conference on Information Systems.* 2005.

de Moor, Aldo. 2005. Patterns for the Pragmatic Web. [ed.] Frithjof and Mugnier, Marie-Laure and Stumme, Gerd Dau. *Conceptual Structures: Common Semantics for Sharing Knowledge.* Berlin, Heidelberg : Springer Berlin Heidelberg, 2005, pp. 1-18.

Dekleva, Sasa and Drehmer, David. 1997. Measuring software engineering evolution: A Rasch calibration. *{Information Systems Research.* 1997, Vol. 8, 1, pp. 95-104.

Duane, Aidan Maurice and OReilly, Philip. 2012. A conceptual stages of growth model for managing an organization's social media business profile (SMBP). 2012.

Evripidou, E. Tamani and P. 2007. A Pragmatic Methodology to Web Service Discovery. *IEEE International Conference on Web Services.* Salt Lake City : IEEE, 2007, pp. 1168-1171.

Facchini, Francesco and Oleśków-Szłapka, Joanna and Ranieri, Luigi and Urbinati, Andrea. 2020. A Maturity Model for Logistics 4.0: An Empirical Analysis and a Roadmap for Future Research. *Sustainability.* 2020, Vol. 12.

Fath-Allah, Abdoullah and Cheikhi, Laila and Al-Qutaish, Rafa E and Idri, Ali. 2015. E-Government Portals Maturity Models: A Best Practices' Coverage Perspective. *Journal of Software.* 7, 2015, Vol. 10.

Fewell, Suzanne and Clark, Thea. 2003. Organisational Interoperability: Evaluation and Further Development of the OIM Model. 2003.

Field, Andy. 2013. *Discovering Statistics Using IBM SPSS Statistics.* 4. s.l. : Sage Publications Ltd, 2013. 1446249182.

Ganzarain, Jaione and Errasti, Nekane. 2016. Three stage maturity model in SME's toward industry 4.0. *Journal of Industrial Engineering and Management (JIEM).* 9, 2016, Vol. 9.

Guédria, Wided and Naudet, Yannick and Chen, David. 2015. Maturity model for enterprise interoperability. *Enterprise Information Systems.* 2015, Vol. 9, 1, pp. 1-28.

Kerzner, Harold. 2019. Using the project management maturity model: strategic planning for project management. 2019, Wiley.

Knight, MR and Widergren, SE and Khandekar A and Kolln, JK and Narang D and Nordman, D. 2020. *Interoperability Maturity Model - A Qualitative and Quantitative Approach for Measuring Interoperability.* São Luiz do Maranhão : U.S. Department of Energy, 2020. p. 48.

Lahrmann, Gerrit and Marx, Frederik and Mettler, Tobias and Winter, Robert and Wortmann, Felix. 2011. Inductive Design of Maturity Models: Applying the Rasch Algorithm for Design Science Research. [ed.] Hemant and Sinha, Atish P and Vitharana, Padmal Jain. *Service-Oriented Perspectives in Design Science Research.* 2011, pp. 176-191.

Leal, Gabriel da Silva Serapião and Guédria, Wided and Panetto, Hervé. 2019. Interoperability assessment: A systematic literature review. *Computers in Industry.* 2019, Vol. 1, pp. 111-132.

Lee, Jejung and Lee, Yugyung and Shah, Sanket and Geller, James. 2007. HIS-KCWater: Context-Aware Geospatial Data and Service Integration. [ed.] Association for Computing Machinery. *Proceedings of the 2007 ACM Symposium on Applied Computing.* New York, NY, USA : s.n., 2007, pp. 24-29.

Lehmkuhl, Tobias and Baumo, Ulrike and Jung, Reinhard. 2013. Towards a maturity model for the adoption of social media as a means of organizational innovation. *2013 46th Hawaii International Conference on System Sciences.* s.l. : IEEE, 2013.

Liu, Shixiong and Li, Weizi and Liu, Kecheng. 2014. Pragmatic Oriented Data Interoperability for Smart Healthcare Information Systems. *2014 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing.* 2014, pp. 811-818.

Liu, Xiufeng. 2010. Using and developing measurement instruments in science education: A Rasch modeling approach. 2010.

Messom, Sulaiman Abdallah Alsheibani and Yen Ping Cheung and Christopher H. 2019. Towards An Artificial Intelligence Maturity Model: From Science Fiction To Business Facts. *PACIS.* 2019.

Mettler, Tobias and Rohner, Peter and Winter, Robert. 2009. Towards a Classification of Maturity Models in Information Systems. *Management of the Interconnected World.* 1 de 2009, pp. 333-340.

Mettler, Tobias. 2010. Supply management im Krankenhaus: Konstruktion und Evaluation eines konfigurierbaren Reifegradmodells zur zielgerichteten Gestaltung. 2010.

Monteiro, Erasmo and Magalhães, Ana Patrícia F. Magalhães Mascarenhas and Maciel, Rita Suzana Pitangueira. 2020. Towards a Methodology for Maturity Models Development: an Exploratory Study in Software Systems Interoperability Domain. [ed.] SBC. *Anais do XIX Simpósio Brasileiro de Qualidade de Software.* 2020, pp. 224-233.

Motta, Rebeca Campos and De Oliveira, Káthia Marçal and Travassos, Guilherme Horta. 2017. Rethinking interoperability in contemporary software systems. *IEEE/ACM Joint 5th International Workshop on Software Engineering for Systems-of-Systems and 11th Workshop on Distributed Software Development, Software Ecosystems and Systems-of-Systems (JSOS).* 2017, pp. 9-15.

Neiva, Frâncila Weidt and David, José Maria N. and Braga, Regina and Campos, Fernanda. 2016. Towards Pragmatic Interoperability to Support Collaboration. 04 de 2016, Vol. 72, C.

Qushem, Umar Bin and Zeki, Akram M and Abubakar, Adamu and Akleylek, Sedat. 2017. The trend of business intelligence adoption and maturity. *International Conference on Computer Science and Engineering (UBMK).* 2017, IEEE.

Rezaei, Reza and Chiew, Thiam Kian and Lee, Sai Peck and Aliee, Zeinab Shams. 2014. Interoperability evaluation models: A systematic review. *Computer in Industry.* 2014, Vol. 1, pp. 1-23.

Ribeiro, Elivaldo Lozer Fracalossi and Monteiro, Erasmo Leite and Claro, Daniela Barreiro and Maciel, Rita Suzana Pitangueira. 2019. A Conceptual Framework for Pragmatic Interoperability. *Proceedings of the XV Brazilian Symposium on Information Systems.* 2019, pp. 36:1--36:8.

Romesburg, C. 2004. *Cluster Analysis for Researchers.* 2004. 9781411606173.

Santos, Kécia Souza Santana and Pinheiro, Larissa Barbosa Leoncio and Maciel, Rita Suzana Pitangueira. 2021. Interoperability Types Classifications: A Tertiary Study. *XVII Brazilian Symposium on Information Systems.* 2021.

Sinderen, C. H. Asuncion and M. J. van. 2010. Pragmatic Interoperability: A Systematic Review of Published Definitions. *Enterprise Architecture, Integration and Interoperability.* Berlin : Springer, 2010, pp. 164-175.

Tolk, Andreas. and Diallo, Saikou Y. and Turnitsa, Charles. 2007. Applying the Levels of Conceptual Interoperability Model in Support of Integratability, Interoperability. *Systemics, Cybernetics and Informatics.* 2007, Vol. 5, 5.

Vivek, Kale. 2019. *Digital Transformation of Enterprise Architecture.* s.l. : CRC Press, 2019. 1138553786.

Wang, W. Wang and A. Tolk and W. 2009. The Levels of Conceptual Interoperability Model. *2009 Spring Simulation Multiconference.* San Diego : SCS, 2009, pp. 168:1-168:9.

Wohlin, Claes and Runeson, Per and Hst, Martin and Ohlsson, Magnus C. and Regnell, Bjrn and Wessln, Anders. 2012. *Experimentation in Software Engineering.* s.l. : Springer Publishing Company, Incorporated, 2012. 3642290434.

Author's contribution

This work is part of Erasmo Leite phd results having Rita Suzana Pitangueira Maciel as advisor. Ana Patricia Magalhães Mascarenhas helped the validation design. Regarding the text, the writing was collaborative. Erasmo Leite started writing the first version of the text. Then, Rita Suzana Pitangueira Maciel and Ana Patricia Magalhães Mascarenhas revised it, performing the general alignment, detailing parts of the text that needed more information, inserting analyzes and criticisms.

Funding Declaration

This study was supported by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) - Grant number 1555173 and Finance Code 001

Data Availability Statement

Data sharing not applicable to this article as no datasets were generated or analysed during the current study.